Short-term forecasting of residential electricity demand using CNN-LSTM

Short-term Forecasting of Residential Electricity Demand Using CNN-LSTM

Ashkan Lotfipoor[1], Sandhya Patidar[2], David Jenkins[2]
[1]PhD researcher, Institute for Infrastructure and Environment, Heriot-Watt University
[2]Associate Professor, Institute for Infrastructure and Environment, Heriot-Watt University

Abstract

Near accurate forecasting of energy demand has become a non-trivial requirement for developing effective management and planning strategies/policies for a resilient energy system. This paper is aimed to develop a novel deep learning-based energy demand prediction model by utilising the combination of Convolutional neural networks and Long Short-term Memory units. The proposed model consists of two one dimensional convolutional layer with max pooling, two bidirectional LSTM layers and finally three fully connected dense layer. The energy consumption data available for a household based in Findhorn ecovillage located in the north of Scotland for a six-week period during the February and March of 2015 was utilised for training, validating, and testing the models. The proposed model provides energy demand prediction for short-term forecasting (5 minutes). The results obtained from the model are compared against four of the classical and widely applied algorithms for time series forecasting: autoregressive integrated moving average (ARIMA), light gradient boosting machine (LightGBM), random forest (RF), and deep neural networks (DNN). The result obtained demonstrated the efficiency of the proposed architecture in outperforming all well-established models.

Introduction

During the past decade, smart meters and grids have become widespread around the globe; for example, the numbers of smart meters installed in the UK reached 14.9 million at the end of June 2019 (Kerai, 2019). Smart meters provide a vast amount of information/data on energy supply and demand to researchers and industry stakeholders.

Over the years, researchers have used this data to improve energy efficiency and sustainability in many frontiers, such as community energy modelling, energy management and energy forecasting. Possibility of near accurate forecasting of energy demand could serve several purposes including optimising the utilisation of available resources and identification of any potential risks, which is essential for developing effective management and planning strategies/policies for a robust and resilient energy system. ASHRAE breaks energy estimation models into two main categories: physics-based and data-driven models (Owen et al., 2009). Data-driven models can be divided into two types of methodologies, a statistical model or a machine learning algorithm.

In recent years, with the boom of artificial intelligence in various fields and the widespread use of machine learning algorithms, many researchers have started testing the potential application of these techniques for accurate forecasting of energy demand. For example, Robinson et al. (2017) in a comprehensive study evaluating the performance of linear regressor, RBF kernel support vector regressor (SVR), AdaBoost regressor, bagging regressor, gradient boosting regressor, random forest, multilayer perceptron regressor (MLP regressor), and K-nearest neighbour regressor. Theile et al. (2018) studied the performance of Support Vector Machine (SVM) and Recurrent Neural Networks on the day-ahead electricity consumption prediction of a group of households. The training of the models was performed using observed weather data while the forecasting was performed using predicted weather data. Johannesen et al. (2019) compared three algorithms, namely Random Forest, K-nearest neighbour and Linear Regressor for forecasting electricity demand profiles of the urban area. Similarly, Huang et al. (2019) used XGboost, Extreme learning machine (ELM) and MLP for energy demand prediction for residential buildings.

The deep learning has also drawn the attention of many researchers working in the area of energy demand prediction. Some related examples include the work of Amarasinghe et al. (2017). They investigated the effectiveness of Convolutional Neural Network (CNN) for performing energy load forecasting at the individual building level. Paterakis et al. (2017) compared the performance of MLPs with SVMs, Gaussian Processes, Regression Trees, Ensemble Boosting and Linear Regression. Kim and Cho (2019a) provided a review of using deep learning with state explainable autoencoder for prediction of electricity consumption profiles. An autoencoder is a neural network that is trained to attempt to copy its input to its output. It learns efficient data representations (encoding) and produces an encoded vector representing the input sequence. This vector is then provided as an input to the decoder model that interprets it, and the output sequence is generated (Goodfellow et al., 2016). Moustris et al. (2020) explored the potential of MLP for the medium, short and very short-term forecasting of electricity load. This study utilises, Human thermal comfort-discomfort biometeorological index as one of the critical features in forecasting. Kim and Cho (2019b) proposed a CNN-LSTM model using the combination of CNN and LSTM for predicting electric energy consumption on IHEPC dataset. Similarly, Le et al. (2019) used a CNN and Bi-LSTM net to predict energy consumption on IHEPC dataset.

Despite this recent development in the previously studied methods for energy demand prediction, little progress has been made in the robustness of these models. Many accurate machine learning models are comparable or even superior to traditional methods, but they have never been deployed as a solution for the public sector. Often, researchers use datasets that are expensive and complex to collect, so when stakeholders want to use these methods for solving their problem, they face the challenge of running these models. To bridge the aforementioned gap, this paper aims to develop a prediction model which is easy to implement and yet is capable of very accurate prediction. The other objectives are to compare the developed methodology with existing methods.

The proposed model provides energy demand prediction (EDP) for short-term forecasting (5 minutes). The results obtained from the model are compared against four of the classical and widely applied algorithms: autoregressive integrated moving average (ARIMA), random forest, light gradient boosting machine (LightGBM), and deep neural networks (DNN). The remaining of the paper is organised as follows: Study area and data collection section present details about the dataset; Methodology section introduces the different elements of the methodology and approach for combining workflow; Result and Discussion section reports the key findings along with a critical discussion on overall success and potentials of proposed methods and ideas for future works.

Study Area and Data Collection

To train and test the developed models, high-resolution data gathered from the Findhorn Ecovillage site, located in northern Scotland, is used in this study. The demand data of Findhorn was collected as part of the Orchestration of renewable integrated generation in neighbourhoods (ORIGIN) project. For the case study, high-resolution data (five-minute interval) collected over a continuous period from February 2015 to March 2015 is used.

The selected case-study household for this study is built on two floors with a total area of approximately 96 m^2. The reading collected from the smart meter is solely related to the overall electricity usage, as the house uses a gas boiler for heating purposes. The dynamics of the electricity demand profile for the case study dwelling during the stated period is presented in figure 2.

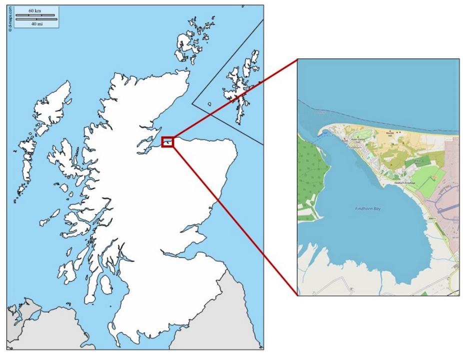

Figure 1: Findhorn Ecovillage Position in Scotland

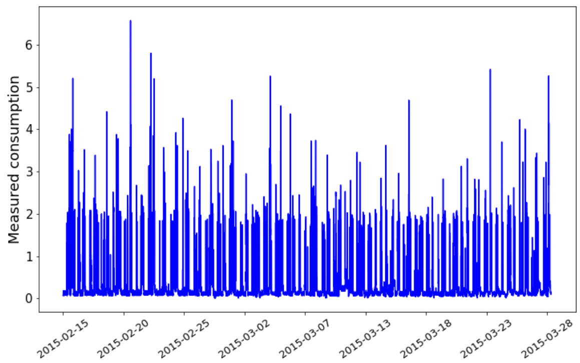

Figure 2: Demand graph from Feb 2015 to Mar 2015

Methodology

Deep Neural Networks

Perceptrons were developed in the 1950s and 1960s by Rosenblatt (1962), inspired by earlier work by McCulloch and Pitts (1943). A perceptron takes in user-specified number of input units, $x_1, x_2, \ldots, x_n$, and each of these units is associated with a specific weight, $w_1, w_2, \ldots, w_n$. These weights are usually multiplied to input units and then summed together to produce the logit of the neuron. The logit is then passed through a linear function to produce the output (Buduma and Locascio, 2017). To learn complex relationships, we need to use neurons that employ some nonlinearity. Today, it is more common to use other types of neurons, namely the sigmoid neuron. A sigmoid neuron is similar to a perceptron, except that the output is not 0 or 1. Instead, it's $\sigma(wx + b)$, where σ is a sigmoid function.

A standard neural network consists of many neurons, each producing a sequence of real-valued activation. Many researchers had wanted for decades to train deep multilayer neural networks, but no successful attempts were reported before 2006 (Bengio, 2009). Hinton et al. (2006) at the University of Toronto introduced Deep Belief Networks (DBNs) with a learning algorithm that greedily trains one layer at a time. Deep learning methods aim at learning feature hierarchies with features from higher levels of the hierarchy formed by the composition of lower-level features (Bengio, 2009). Deep feedforward networks, also called feedforward neural networks, or multilayer perceptrons (MLPs), are the quintessential deep learning models (Goodfellow et al., 2016).

Convolutional Neural Networks

Convolutional nets were inspired by the visual system's structure. The first computational models based on these local connectivities between neurons and hierarchically organised transformations of the image are found in Fukushima (1980). Later, LeCun and collaborators designed and trained convolutional networks using the error gradient, obtaining state-of-the-art performance (LeCun et al., 1998).

Convolution preserves the relationship between pixels by learning image features using small squares of input data. It is a mathematical operation that takes two inputs, such as an image matrix and a filter.

The architecture of a CNN is designed to take advantage of the 2D structure of an input image. This is achieved with local connections and tied weights followed by some form of pooling (Average/Maximum) which results in translation-invariant features. Similarly, to computer vision tasks, in time series problems it is desired to extract a small number of low-level features with a small receptive field across the entire input. This method can significantly improve the accuracy of prediction system while keeping the computation cost at an acceptable range.

Long Short-term Memory

The most effective sequence models in deep learning are called gated recurrent neural networks. These include the long short-term memory (LSTM) and the gated recurrent units (GRU) (Goodfellow et al., 2016). To combat the problem of vanishing gradients (Hochreiter et al., 2001), Hochreiter and Schmidhuber (1997) introduced the LSTM architecture by introducing self-loops to produce paths where the gradient can flow for a long duration. A crucial addition has been to make the weight on this self-loop conditioned on the context, rather than fixed (Gers et al., 2000).

LSTMs make small modifications to the information by multiplications and additions. With LSTMs, the information flows through a mechanism known as cell states, and the cell states allow LSTMs to let information through selectively. They are composed out of a sigmoid neuron and a point-wise product. Those gates act on the signals they receive, and similar to the neural network's nodes, they block or pass on information based on its strength and import, which they filter with their own sets of weights. Those weights, like the weights that modulate input and hidden states, are adjusted via the learning processes of recurrent networks. That is, the cells learn when to allow data to enter, leave or to be deleted through the iterative process of making guesses, backpropagating error, and adjusting weights via gradient descent (Goodfellow et al., 2016).

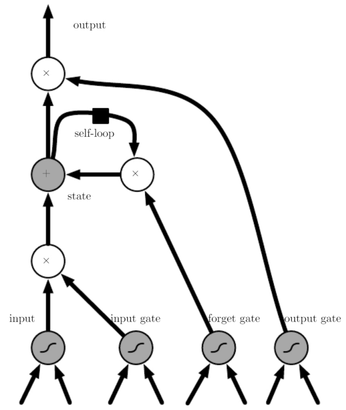

Figure 3: LSTM Architecture (Goodfellow et al., 2016)

Model Architecture

In many recent studies, both CNNs and LSTMs have shown a significant increase in performance over Deep Neural Networks (DNNs) across a variety of tasks (Huang and Kuo, 2018; Ding et al., 2018; Núñez et al., 2018; Zhao et al., 2019; Bhunia et al., 2019). CNNs, LSTMs and DNNs are complementary in their modelling capabilities, as CNNs are good at reducing frequency variations, LSTMs are good at sequence modelling, and DNNs are appropriate for mapping features to a more separable space (Sainath et al., 2015). In this paper, a deep neural net is developed, which will use this complementarity to predict energy demand.

Figure 4 shows the complete architecture of the developed model. The two convolutional layers will extract features from the input variables (energy demand with 5 min interval). The convolutional layers develop a feature map of the input variables which will improve the accuracy of the model in comparison with vanilla LSTM networks. After each convolutional layer, there is a max-pooling layer which acts as a tool to reduce over-fitting and to minimise the computational cost by reducing the number of parameters to learn. Max pooling is a sample-based discretisation process, and the objective is to down-sample an input representation and reducing its dimensionality (Kulkarni and Satapathy, 2019). By using a pooling layer in the architecture, only features with the highest value and importance will be selected.

After flattening, converting a matrix to a single array, the output of convolutional layers, the information is passed to the three Bi-LSTM layers to detect patterns in long periods in both directions. Finally, the output of the layers will pass onto the four fully connected

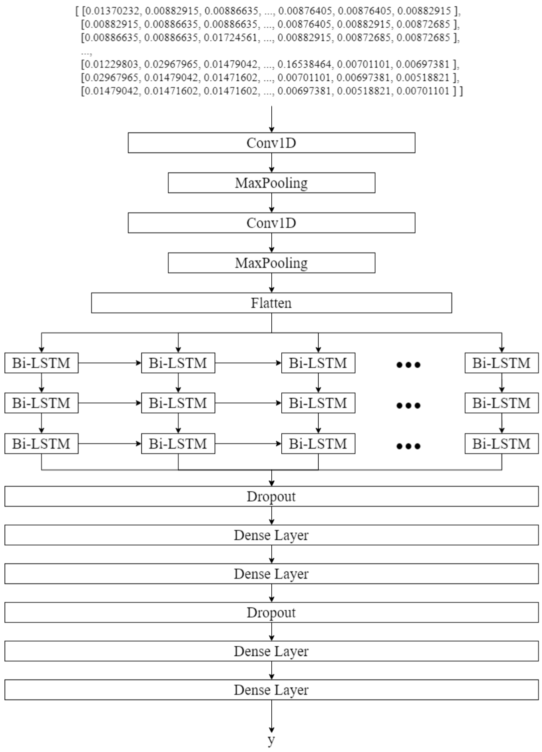

Figure 4: CNN-LSTM Architecture

dense layers with one dropout layer in between for making a prediction.

The proposed methodology was implemented to perform prediction for the next time step of (five-minute) consumption. In order to do this, 35 previous observation was fed into the model as lagged features. This number was selected based on a grid search to obtain the optimum window size for maximising the performance and minimise the computation cost. It worth mentioning that the predicted value is not used for forecasting the next time step. For the CNN layers, the filter size of 128 was selected as the optimum number with a kernel size of 2. The activation function for these layers was the linear rectifier unit (ReLU) function which is the default recommendation in modern neural networks (Goodfellow et al., 2016). As explained earlier for pooling phase of the network, the max-pooling was selected with a pool size of 2. Once the CNN layers made their feature maps, and it was flattened, their output was sent to two Bi-LSTM layers.

Bidirectional RNN is consist of two independent RNNs together. This structure allows the networks to have both backward and forward information about the sequence at each time step. Bidirectional layers will run the inputs in two ways, one from past to future and one from future to past and what makes this method powerful is that it enables the network in any point in time to preserve information from both past and future. Based on our experiments, using this architecture help the model to better understand the patterns in the demand load. Next, for regularising the network, a dropout layer with a dropout rate of 0.3 was used after the second Bi-LSTM layer.

In the developed model, three fully connected dense layer with 50, 10 and 1 neurons were used, respectively. These layers used the ReLU function as their activation function. For compiling the model, Adaptive Moment Estimation (Adam) (Kingma and Ba, 2014) algorithm was used as the gradient-based optimiser. The value of the learning rate was obtained by running the model first with a learning rate scheduler. Mean square error (MSE) was used as the loss function for training the model and root mean square error (RMSE) and mean absolute error (MAE) were used as the metrics for evaluating the performance of the model. Furthermore, an early stopping method was used to prevent the model from overfitting on the train data. The model was trained in 200 epochs with a batch size of 256. In total, 1,385,275 parameters were trained during the model development phase. The model was developed using the Keras neural-network library with TensorFlow 2.1.0 backend in Python.

Evaluation index of model performances

The predicted values by the proposed model are evaluated by three performance metrics for regression models, MAE, MSE and RMSE. Furthermore, the residual error was calculated for test data to evaluate the performance of the model by plotting its distribution. In regression analysis, the difference between the observed value of the dependent variable (y_i) and the predicted value (x_i) is called the residual error (e).

$$e = y_i - x_i$$

MAE measures the average magnitude of the errors in predicted values, without considering their direction. The MAE is a linear score which means that all the individual differences are weighted equally in the average. MAE is calculated as follows.

$$MAE = \frac{1}{n}\sum_{i=1}^{n} abs(y_i - x_i)$$

MSE measures the average of the squares of the errors. In other words, it is the average squared difference between the predicted values and the actual values. The equation for MSE is as follows.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - x_i)^2$$

RMSE is the standard deviation of the residual errors. It is the square root of the average of squared differences between predictions and actual values. The RMSE is calculated as:

$$RMSE = \sqrt{(\frac{1}{n})\sum_{i=1}^{n}(y_i - x_i)^2}$$

Table 1: Performance of different algorithms on the dataset

Evaluation Results												
	ARIMA		*Random Forest*		*LightGBM*		*DNN*		*CNN-LSTM*		*C24*	
	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test
MAE	0.230	-	0.207	0.205	0.245	0.276	0.202	0.201	0.200	0.192	-	0.180
MSE	0.227	-	0.200	0.298	0.154	0.225	0.211	0.209	0.206	0.214	-	0.151
RMSE	0.477	-	0.459	0.476	0.493	0.475	0.479	0.469	0.454	0.463	-	0.389

Results and Discussion

The performance of the developed network is evaluated against one mainstream statistical method (ARIMA) and three established machine learning algorithms, namely Random Forest, Light Gradient Boosting Machine (LightGBM) (Ke et al., 2017) and a DNN. LightGBM is a gradient boosting framework that uses tree-based learning algorithms. For the ARIMA model, the values of the hyperparameters (p, d, q) were chosen by using autocorrelation function (ACF) and partial autocorrelation function (PACF). For Random Forest and LightGBM model, the values of the hyperparameters were tuned by performing grid search so that the maximum performance of the individual models were achieved. Moreover, cross-validation was performed to check the degree of generalisation of the models. For the DNN, four dense layers with 15, 10, 5 and 1 neurons were developed, and similar values for hyperparameters (Activation function = ReLU, Optimisation algorithm = Adam with a learning rate of 1e-4) were used. All models used the same 80/20 ratio for splitting the dataset into training and testing set; however, the train/validation/test configuration was used in the development of the neural networks (20% of train data was used as validation set). Table 1 summaries the results (MAE, MSE and RMSE) obtained from all the algorithms for training and testing sets.

Although the difference in the performance of different models is not substantial, it can be seen that the CNN-LSTM network provides the lowest values in all three evaluation metrics for an accurate forecast, outperforming all other models. DNNs require a large size training data because of the considerable number of parameters needed to be tuned by a learning algorithm. The issue in DNN is that the network starts with a poor initial state and then an optimisation algorithm such as stochastic gradient descent or Adam is used to converge the network to an optimal position. So, given more data to the CNN-LSTM network, it is likely it outperform other models significantly.

Also, it can be argued that the CNN-LSTM network can be generalised best to unseen data due to the low difference between RMSE for train and test data, which increase the robustness and generality of the

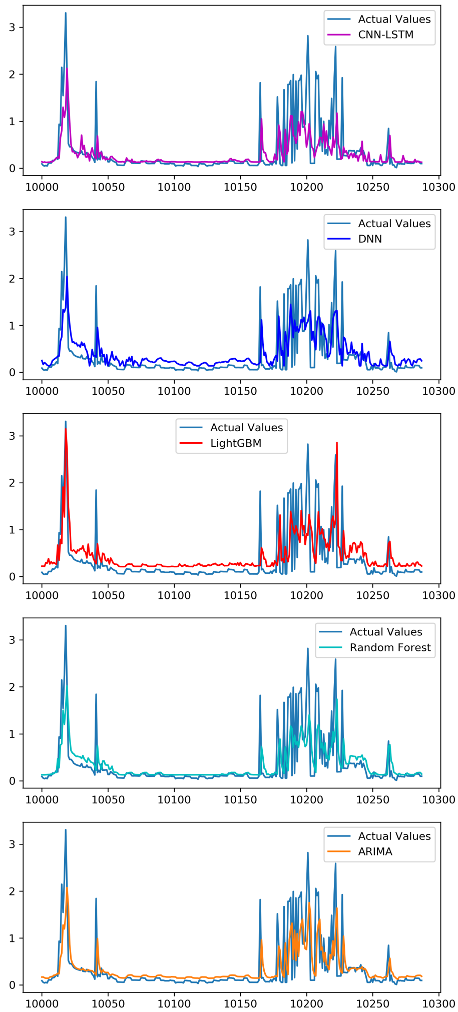

Figure 5: Predicted values and actual values for a day

model. The reason for this could be all the measures that were implemented to prevent the model from overfittings, such as the use of max-pooling layers, dropout layer, and early stopping. Furthermore, it is worth noting that all machine learning algorithms outperform the statistical model used in this study, showing substantial promise for the machine learning algorithms in time series forecasting of electricity demand load.

Figure 5 depicts a prediction from all models against the actual measurement in the testing set. As it can

be seen, RF and CNN-LSTM can follow the general trend of data showing capabilities of generalisation. ARIMA model also predicts accurately, however it has seen the test data during model development. If the model sees the test data during the model development phase, its accuracy and generalisation cannot be adequately assessed.
Although the most commonly applied metrics were used here to assess the accuracy of model predictions, it is particularly difficult to conclude whether the predictions are consistently accurate or not. For this reason, the probability density function (PDF) of residual errors, depicted in Figure 6, was created to check the distribution of the error. A symmetrically distributed probability density functions with mean closer to zero, and small standard deviation produces more accurate and reliable results. The PDF graph of CNN-LSTM has the closest mean to zero than any other method.

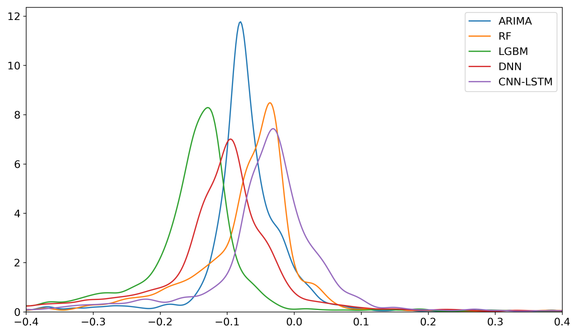

Figure 6: Histogram of Residual Error on test data

Furthermore, to evaluate the efficiency of the developed model in prediction of unseen data, a different dwelling from Findhorn village was selected to perform further testing of the model. It is worth noting that the chosen building has similar construction and overall design as the building used for training the model. The result shows (under C24 in Table 1), even better performance of the developed model on this dataset. The proposed model predicted the demand for this building by RMSE value of 0.389.
As mentioned earlier, the CNN-LSTM network outperforms other approaches, however, the method needs to be tested on several different datasets with many different architectures (number of neurons, layers, etc.) to accurately analyse the effectiveness of this algorithm for generating near accurate energy demand prediction.

Conclusion

This study was conducted to investigate the performance of a CNN-LSTM model for generating energy demand forecasting for residential buildings. The presented model was trained and tested on demand load of a building in Findhorn ecovillage in Scotland and then further validated through an independent application on another building in the same area. The novel structure of the proposed model consists of two one-dimensional convolutional layers with max-pooling, three bidirectional LSTM layers and three fully connected dense layer. Furthermore, the performance of the model was tested against ARIMA, RF, LightGBM and a DNN to see if CNN-LSTM show any advantage in this problem. The result obtained demonstrated the efficiency of the proposed architecture in outperforming all well-established models. The RMSE value measured on the test data for CNN-LSTM was significantly lower than the rest of the other algorithms investigated herein. Thus, it can be concluded that for the present investigation, CNN-LSTM architecture shows a good performance in generating accurate load forecasting. There are needs for a detailed investigation to explore further potentials of the proposed model. For the future work, the authors plan to implement a multivariate approach to improve the load forecasting abilities further and to use a variety of features such as weather-related data and building information in parallel. In addition to this, it would be interesting to see if the addition of more dataset in training and evaluation steps of the model development could contribute to achieving an even better performance. Furthermore, future works will focus on the real-world problem of multi-step forecasting for household electricity consumption.

Acknowledgement

The authors would like to acknowledge the Findhorn Foundation and the academic and industry partners within the Centre for Energy Systems Integration (CESI - National Centre for Energy System Integration).

References

Amarasinghe, K., D. L. Marino, and M. Manic (2017). Deep neural networks for energy load forecasting. In *2017 IEEE 26th International Symposium on Industrial Electronics (ISIE)*, pp. 1483–1488.

Bengio, Y. (2009). Learning deep architectures for ai. *Foundations and trends® in Machine Learning 2*(1), 1–127.

Bhunia, A. K., A. Konwer, A. K. Bhunia, A. Bhowmick, P. P. Roy, and U. Pal (2019). Script identification in natural scene image and video frames using an attention based convolutional-lstm network. *Pattern Recognition 85*, 172–184.

Buduma, N. and N. Locascio (2017). *Fundamentals of Deep Learning: Designing Next-Generation Machine Intelligence Algorithms*. O'Reilly Media, Inc.

Ding, L., W. Fang, H. Luo, P. E. D. Love, B. Zhong, and X. Ouyang (2018). A deep hybrid learning model to detect unsafe behavior: Integrating convolution neural networks and long short-term memory. *Automation in Construction 86*, 118–124.

Fukushima, K. (1980). Neocognitron: a self organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biol Cybern 36*(4), 193–202.

Gers, F. A., J. A. Schmidhuber, and F. A. Cummins (2000). Learning to forget: Continual prediction with lstm. *Neural Comput. 12*(10), 2451–2471.

Goodfellow, I., Y. Bengio, and A. Courville (2016). *Deep Learning*. MIT Press.

Hinton, G. E., S. Osindero, and Y.-W. Teh (2006). A fast learning algorithm for deep belief nets. *Neural computation 18*(7), 1527–1554.

Hochreiter, S., Y. Bengio, P. Frasconi, and J. Schmidhuber (2001). *Gradient flow in recurrent nets: the difficulty of learning long-term dependencies*, Volume 6, pp. 237–243. IEEE Press.

Hochreiter, S. and J. Schmidhuber (1997). Long short-term memory. *Neural Computation 9*(8), 1735–1780.

Huang, C. J. and P. H. Kuo (2018). A deep cnn-lstm model for particulate matter (pm2.5) forecasting in smart cities. *Sensors (Switzerland) 18*(7).

Huang, Y., Y. Yuan, H. Chen, J. Wang, Y. Guo, and T. Ahmad (2019). A novel energy demand prediction strategy for residential buildings based on ensemble learning. In *Energy Procedia*, Volume 158, pp. 3411–3416.

Johannesen, N. J., M. Kolhe, and M. Goodwin (2019). Relative evaluation of regression tools for urban area electrical energy demand forecasting. *Journal of Cleaner Production 218*, 555–564.

Ke, G., Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu (2017). Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in neural information processing systems*, pp. 3146–3154.

Kerai, M. (2019). Statistical release and data: Smart meters, great britain, quarter 4 2019. Report, Department for Business, Energy and Industrial Strategy.

Kim, J. Y. and S. B. Cho (2019a). Electric energy consumption prediction by deep learning with state explainable autoencoder. *Energies 12*(4), 739.

Kim, T.-Y. and S.-B. Cho (2019b). Predicting residential energy consumption using cnn-lstm neural networks. *Energy 182*, 72–81.

Kingma, D. P. and J. Ba (2014). Adam: A method for stochastic optimization.

Kulkarni, A. and S. Satapathy (2019). *Optimization in Machine Learning and Applications*. Springer Singapore.

Le, T., M. T. Vo, B. Vo, E. Hwang, S. Rho, and S. W. Baik (2019). Improving electric energy consumption prediction using cnn and bi-lstm. *Applied Sciences (Switzerland) 9*(20).

LeCun, Y., L. Bottou, Y. Bengio, and P. Haffner (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE 86*(11), 2278–2324.

McCulloch, W. S. and W. Pitts (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics 5*(4), 115–133.

Moustris, K., K. A. Kavadias, D. Zafirakis, and J. K. Kaldellis (2020). Medium, short and very short-term prognosis of load demand for the greek island of tilos using artificial neural networks and human thermal comfort-discomfort biometeorological data. *Renewable Energy 147*, 100–109.

Núñez, J. C., R. Cabido, J. J. Pantrigo, A. S. Montemayor, and J. F. Vélez (2018). Convolutional neural networks and long short-term memory for skeleton-based human activity and hand gesture recognition. *Pattern Recognition 76*, 80–94.

Owen, M., R. American Society of Heating, and A.-C. Engineers (2009). *2009 ASHRAE Handbook: Fundamentals*. American Society of Heating, Refrigeration and Air-Conditioning Engineers.

Paterakis, N. G., E. Mocanu, M. Gibescu, B. Stappers, and W. v. Alst (2017). Deep learning versus traditional machine learning methods for aggregated energy demand prediction. In *2017 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*, pp. 1–6.

Robinson, C., B. Dilkina, J. Hubbs, W. Zhang, S. Guhathakurta, M. A. Brown, and R. M. Pendyala (2017). Machine learning approaches for estimating commercial building energy consumption. *Applied Energy 208*, 889–904.

Rosenblatt, F. (1962). *Principles of neurodynamics: perceptrons and the theory of brain mechanisms*. Spartan Books.

Sainath, T. N., O. Vinyals, A. Senior, and H. Sak (2015). Convolutional, long short-term memory, fully connected deep neural networks. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4580–4584.

Theile, P., A. L. Towle, K. Karnataki, A. Crosara, K. Paridari, G. Turk, and L. Nordstrom (2018). Day-ahead electricity consumption prediction of a population of households: Analyzing different machine learning techniques based on real data from rte in france. In *2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids, SmartGridComm 2018*.

Zhao, J., X. Mao, and L. Chen (2019). Speech emotion recognition using deep 1d and 2d cnn lstm networks. *Biomedical Signal Processing and Control 47*, 312–323.